REMARKS

This Response is submitted in response to the final Office Action of October 18, 2004 (hereinafter "the Office Action"). The due date for response extends to January 18, 2005. Applicant respectfully requests reconsideration of the outstanding rejection in view of the following Remarks.

All references to the claims, except as noted, will be made with reference to the claim list provided in the claim list in the Amendment submitted August 13, 2004. All references to "the Office Action," except as noted, will be referencing the most recent Office Action dated October 18, 2004. Line numbers in the Office Action, except as noted, will count every printed line, except the page header, but including section headings. Explanations of prior art references are based on the undersigned's best understanding thereof. If there is any confusion or questions regarding any aspect of this Amendment, the Examiner is invited to contact the undersigned.

Status

Applicant notes with appreciation the withdrawal of the rejections under 35 U.S.C. §§ 112 and 101. Claims 1-8 and 17-24 stand finally rejected under 35 U.S.C. § 103(a).

Rejection under 35 U.S.C. §103(a)

Claims 1-8 and 17-24 are finally rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent 6,496,871 to Jagannathan et al. (Jagannathan) in view of the article entitled, "Migration of Processes, Files, and Virtual Devices in the MDX Operating System" by Schrimpf (Schrimpf). Applicant respectfully traverses because the prior art does not teach or suggest every feature set forth in the claims and because there was no motivation to combine the references at the time the invention was made.

1. *Not All Limitations Taught or Suggested By Prior Art*

Claim 1 sets forth, *inter alia*, "encapsulating a plurality of interconnected processes into a compute capsule . . . ; encapsulating a system environment interconnected with said interconnected processes into said compute capsule. . . ." The office action, page 2, line 21, suggests that Jagemnathan teaches encapsulation of a system environment. Applicant respectfully disagrees.

Jagannathan teaches a programming tool that allows computer programmers to easily create distributed and movable objects using agents to provide a shared memory abstraction in a distributed networked environment (col. 9, lines 24-26). The "state information" referenced in the Office Action (in the bottom paragraph of page 2 of the Office Action) is in reference to the agents which provide the object space in which the distributed objects exist, and this information is not encapsulated. Furthermore, while Jagannathan does mention "encapsulation" it does not have the same meaning as "encapsulation" as presently disclosed. Specifically, Jagannathan talks of encapsulation as placing an object in a "protection domain" (col. 8, lines 65-68). This is different from the meaning of "encapsulation" as presently disclosed, which is directed to a representation of processes and associated state.

Schrimpf does not overcome the deficiencies of Jagannathan. Specifically, Schrimpf teaches an operating system with the ability to promote load balancing and sharing among a plurality of processors in a distributed system by migrating existing processes. However, Schrimpf does not mention encapsulating a process with its environment, nor encapsulating a plurality of processes. Thus, neither Jagannathan nor Schrimpf teach encapsulating a system environment. While Schrimpf does teach copying data related to a process and its state, no mention is made of encapsulation or any analog thereof.

Furthermore, Applicant respectfully submits that Schrimpf implicitly teaches away from encapsulation. First, creating a compute capsule representing an active computing environment would not necessarily aid in load balancing if that environment represented a great deal of load on system resources, since the entirety of that load would simply be placed onto another resource and the load would still be unbalanced. Secondly, the encapsulation steps would add to the number of steps and therefore the time it would take to migrate the processes. Schrimpf explicitly teaches that migration should take as little time as possible. See, e.g., page 72, lines 4-5 of the first full paragraph: "Migration . . . should not interrupt

execution for long." Thus, Schrimpf implicitly teaches away from encapsulation as it would have increased the time necessary to make the migration and it would not necessarily have been helpful in load balancing.

Since neither Jagannathan nor Schrimpf teach or suggest encapsulating a system environment such as set forth in claim 1 and similarly in claim 17, and because all pending rejected claims depend from one of these independent claims, each and every feature set forth in the claims is not met by the cited prior art. Applicant therefore respectfully submits that the rejection against claims 1-8 and 17-24 should be withdrawn.

2. *No Suggestion or Motivation to Combine and/or Modify References*

The Office Action suggests that "it would have been obvious . . . to apply the teaching of determining a state of the capsule and caching the interconnected processes and the state as taught by Schrimpf to the invention of Jagannathan, because this allows the context data structure of the processes to be patched at a destination system and allows the processes to execute at the destination system [p. 77, section 5.3, lines 10-14 of Schrimpf], which provides load balancing in distributed systems to employ all available processors and keep work queues similar in length [p. 70, section 1, lines 1-2 of Schrimpf]" (page 3, lines 10-17 of the office action). Applicant respectfully disagrees that Schrimpf teaches encapsulating system environment information related to encapsulated processes. However, even if, for the sake of argument, Schrimpf did teach such encapsulation, there would have been no motivation to combine with Jagannathan.

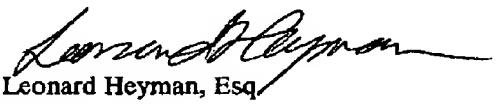
As mentioned above, Schrimpf teaches a system for migrating processes. Jagannathan teaches an agent and system that allows computer programmers to create distributed objects. While both references deal with distributed software, they are not obviously combinable since Jagannathan allows development of application-level software and Schrimpf is an operating system and therefore operates at a much lower level, i.e., closer to hardware. Jagannathan provides a shared memory abstraction layer for computer program developers while Schrimpf provides a low-level operating system that allows for migrating processes for the purposes of load balancing. The specifics of core image or other state information mentioned by Schrimpf (page 72, first paragraph) are not available to Jagannathan, since Jagannathan is directed to agents or "protection domains" which exist on top of an underlying operating system. Even if Jagannathan had access to such information, it would not be able to utilize it.

Therefore, since there was no suggestion or motivation to modify and/or combine Jagannathan and Schrimpf in the manner set forth in the Office Action, Applicant respectfully submits that the rejection against claims 1-8 and 17-24 under 35 U.S.C. § 103(a) which combines Schrimpf and Jagannathan should be withdrawn.

In view of the foregoing Amendments and Remarks, Applicants respectfully submit that the present application is in condition for allowance. A Notice of Allowance is therefore respectfully requested.

If the Examiner has any questions concerning the present amendment, the Examiner is kindly requested to contact the undersigned at (408) 774-6933. If any other fees are due in connection with filing this amendment, the Commissioner is also authorized to charge Deposit Account No. 50-0805 (Order No. SUNMP585). A duplicate copy of the transmittal is enclosed for this purpose.

Respectfully submitted,
MARTINE & PENILLA, LLP



Leonard Heyman, Esq.
Reg. No. 40,418

710 Lakeway Drive, Suite 200
Sunnyvale, CA 94085
Telephone: (408) 749-6900
Facsimile: (408) 749-6901
Customer No. 32291